WE CLAIM:

1. An encryption method for providing both data confidentiality and integrity for a message, comprising the steps of:

receiving an input plaintext string comprising a message and padding it as necessary such that its length is a multiple of ℓ bits;

partitioning the input plaintext string a length that is a multiple of ℓ bits into a plurality of equal-size blocks of ℓ bits in length;

creating an MDC block of ℓ bits in length that includes the result of applying a non-cryptographic Manipulation Detection Code (MDC) function to the plurality of the equal-size blocks;

making one and only one processing pass with a single cryptographic primitive over each of said equal-size blocks and the MDC block to create a plurality of hidden ciphertext blocks each of ℓ bits in length; and

performing a randomization function over said plurality of hidden ciphertext blocks to create a plurality of output ciphertext blocks each of ℓ bits in length.

2. The method as defined in claim 1, comprising the steps of:

wherein said making one and only one processing pass step comprises processing each of said equal-size blocks and the MDC block by an encryption scheme that is confidentiality-secure against chosen-plaintext attacks, wherein each of said equal-size blocks and the MDC block is processed by a block cipher using a first secret key to obtain said plurality of hidden ciphertext blocks; and

wherein said performing a randomization function step comprises combining each of said hidden ciphertext blocks with a corresponding element of a sequence of unpredictable elements to create a set of output blocks of the ciphertext, wherein a hidden ciphertext block identified by an index i is combined with the element of the sequence identified by index i by an operation that has an inverse.

3. The method as defined in claim 2, wherein said creating an MDC block step comprises:

applying the non-cryptographic MDC function to the partitioned plaintext blocks; and

combining the result with a secret, ℓ-bit random vector generated on a per-message basis to obtain said MDC block.

4. The method as defined in claim 3, wherein said combining step comprises performing the combination using an bit-wise exclusive-or function.

5. The method as defined in claim 3, comprising the step of generating said secret random vector from a secret random number generated on a per-message basis.

6. The method as defined in claim 2, further comprising the step of appending the created MDC block after a last block of the set of equal-sized blocks comprising the padded plaintext string.

7. The method as defined in claim 3, wherein said encryption scheme is cipher block chaining (CBC); and further comprising the step of representing an initialization vector for the CBC as the secret random vector.

8. The method as defined in claim 2, wherein the hidden ciphertext blocks from the processing step comprise n + 1 hidden ciphertext blocks each of ℓ-bit length, where n is the total number of blocks in said set of equal-sized blocks of the padded input plaintext string.

9. The method as defined in claim 2, further comprising the step of generating each of a plurality of the unpredictable elements of said sequence of unpredictable elements by combining a different element identifier for each of the unpredictable elements and a secret random number.

10. The method as defined in claim 5, further comprising the step of generating each of a plurality of the unpredictable elements of said sequence of unpredictable elements

by combining a different element identifier for each of the unpredictable elements and said secret random number.

11. The method as defined in claim 5, further comprising the steps of:

enciphering the secret random number using the block cipher using the secret first key; and

including this enciphered secret random number as one of said output ciphertext blocks.

12. The method of claim 3, wherein said secret random vector is generated by enciphering a secret random number of ℓ bits in length, said enciphering using said block cipher using a secret second key.

13. The method as defined in claim 5, wherein said secret random vector is generated by enciphering a variant of said secret random number of ℓ bits in length, said enciphering using said block cipher using said secret first key.

14. The method as defined in claim 13, wherein said variant of said secret random number is obtained by adding a constant to said secret random number.

15. The method of claim 5, wherein the secret random number is provided by a random number generator.

16. The method as defined in claim 5, further comprising:

generating said secret random number by enciphering a count of a counter initialized to a constant, said enciphering being performed with the block cipher using the secret first key; and

incrementing said counter by one on every message encryption.

17. The method as defined in claim 16, wherein said counter is initialized to a constant whose value is the ℓ-bit representation of negative one.

18. The method as defined in claim 16, comprising:

initializing said counter to a secret value of ℓ bits in length.

19. The method as defined in claim 16, further comprising:

outputting said counter value as an output block of the encryption scheme.

20. The method as defined in claim 5, further comprising:

sharing the secret random number between a sender and a receiver.

21. The method as defined in claim 1, wherein said non-cryptographic MDC function is a bit-wise exclusive-or function.

22. The method as defined in claim 2, wherein said encryption scheme is the CBC scheme of encryption.

23. The method as defined in claim 2, wherein said operation that has an inverse is the addition modulo 2^{ℓ}.

24. The method as defined in claim 2, wherein said operation that has an inverse is a bit-wise exclusive-or operation.

25. The method as defined in claim 2, wherein said operation that has an inverse is the subtraction modulo 2^{ℓ} operation.

26. The method as defined in claim 3, further comprising:

generating said secret random vector from a secret random number of ℓ-bit length; and

generating each element in said sequence of unpredictable elements by modular 2^{ℓ} multiplication of a different unique element identifier (i) for each element in the sequence of unpredictable elements and said secret random number.

27. The method as defined in claim 3, further comprising:

generating said secret random vector from a secret random number of ℓ-bit length; and

generating each element in said sequence of unpredictable elements from the previous element by modular 2^{ℓ} addition of said secret random number to the previous element, with a first element of said sequence being said secret random number itself.

28. A decryption method that is the inverse of an encryption method which provides both data confidentiality and integrity, comprising the steps of:

presenting a string including ciphertext string for decryption;

partitioning said ciphertext string into a plurality of ciphertext blocks comprising ℓ bits each;

selecting n+1 ciphertext blocks from said plurality of ciphertext blocks representing n data blocks and one MDC block and performing a reverse randomization function on each of the selected n+1 ciphertext blocks to obtain a plurality of hidden ciphertext blocks each of ℓ bits in length;

making one and only one processing pass with a single cryptographic primitive that is the inverse of an encryption single cryptographic primitive over the plurality of hidden ciphertext blocks to obtain a plurality of plaintext blocks comprising ℓ bits each;

verifying integrity of the plaintext blocks using a non-cryptographic Manipulation Detection Code (MDC) function;

outputting the plurality of plaintext blocks as an accurate plaintext string if the integrity verification passes; and

outputting a failure indicator if the integrity verification fails.

29. The method as defined in claim 28, wherein performing said reverse randomization function comprises:

deriving a secret random number from said ciphertext string presented for decryption;

generating a sequence of unpredictable elements each of ℓ-bit length from said secret random number in a same manner as used at the encryption method;

selecting n+1 ciphertext blocks from said plurality of ciphertext blocks representing n data blocks and one MDC block in a same order as that used at the encryption method, and combining said selected ciphertext blocks with said sequence of unpredictable elements to obtain a plurality of hidden ciphertext blocks, such that each of the n+1 ciphertext blocks identified by index i is combined with the element of the sequence of unpredictable elements identified by index i, by the inverse of an operation used at the encryption method;

wherein the step of making one and only one processing pass comprises decrypting the plurality of hidden ciphertext blocks with the inverse of the block cipher used at an encryption method with a first secret key (K), the result of the decryption being a plurality of n decrypted plaintext data blocks and one decrypted MDC block each of ℓ-bit length; and

wherein the verifying integrity step comprises creating an MDC decryption block by applying the non-cryptographic Manipulation Detection Code function to the n decrypted plaintext data blocks and combining the result with a secret, ℓ-bit random vector, said combining operation being the same as a combining operation at the encryption method, and said secret random vector being derived from said secret random number in the same manner as at the encryption method; and comparing said created MDC decryption block with the decrypted MDC block.

30. The method of claim 28, further comprising:

selecting the ciphertext block of a secret random number from said string presented for decryption; and

deciphering the selected ciphertext block to obtain the secret random number.

31. The method as defined in claim 30, wherein said deciphering step comprises performing the deciphering with the inverse of the said block cipher using the secret first key.

32. The method of claim 29, further comprising:

for the encryption method generating a secret random number by enciphering a count of a counter initialized to a constant, said enciphering being performed with the block cipher using the secret first key; and

incrementing said counter by one on every message encryption; and

further comprising for decrypting the ciphertext blocks of the partitioned ciphertext string the steps of:

selecting a counter block representing the count of the counter from said string presented at decryption; and

enciphering said selected counter block to obtain the secret random number.

33. The method as defined in claim 32, wherein the enciphering step comprises performing said enciphering with the block cipher using the secret first key.

34. The method as defined in claim 28, wherein the string presented for decryption is obtained by applying the encryption method that provides both data confidentiality and integrity to an input plaintext string, further comprising:

outputting said input plaintext string.

35. A method for parallel encryption processing of a message comprising the steps of:

partitioning said input plaintext string into a plurality of input plaintext segments;

concurrently presenting each different one of said plurality of input plaintext segments to a different one of a plurality of encryption processors, each of said different processors using a different ℓ-bit secret random number per segment to obtain a ciphertext segment using an encryption method providing both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, and using a non-cryptographic Manipulation Detection Code function, wherein said single cryptographic primitive is a ℓ-bit block cipher using a secret first key;

assembling the plurality of ciphertext segments into a ciphertext string; and

outputting the ciphertext string.

36. The method as defined in claim 35, wherein said assembling step comprises including in the ciphertext string the number of ciphertext segments, a ciphertext segment index, a length of each ciphertext segment and a sequence of ciphertext segments.

37. The method of claim 35, further comprising:

generating said different ℓ-bit secret random number per segment from a secret random number of ℓ bits in length.

38. The method of claim 37, further comprising:

generating said different secret random number per segment from the secret random number of ℓ bits by adding modulo 2^{ℓ} a plaintext segment sequence index for that segment to the secret random number.

39. The method of claim 37, further comprising:

generating said secret random number of ℓ bits in length by a random number generator;

enciphering said secret random number with said block cipher using a first key; and

including the enciphered secret random number as an output block of said output ciphertext string.

40. The method of claim 37, further comprising:

generating said secret random number of ℓ bits in length by enciphering a counter initialized to a constant, said enciphering being done with said block cipher using said first key; and

outputting said counter value as an output block of said output ciphertext string; and

incrementing after every different message encryption said counter by a number equal to a number of plaintext segments in the message.

41. A method for parallel decryption processing of a message comprising the steps of:

presenting a string including the ciphertext string of a message for decryption;

partitioning said ciphertext string into a plurality of ciphertext segments;

concurrently presenting said plurality of ciphertext segments to a plurality of processors;

obtaining a different secret random number per ciphertext segment from a secret random number in the same manner as at a parallel encryption method;

decrypting each ciphertext segment using said different secret random number per ciphertext segment to obtain a plaintext segment, using a decryption method that is the inverse of an encryption method used in the parallel encryption method that provides both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, wherein said single cryptographic primitive is a ℓ-bit block cipher using a secret first key, and using a non-cryptographic Manipulation Detection Code function for verifying integrity of the plaintext blocks of each plaintext segment;

assembling the plurality of plaintext segments into a plaintext string; and

verifying the integrity of the plaintext segments and their sequence and outputting the plaintext string if the integrity verification passes.

42. The method as defined in claim 41, further comprising outputting a failure indicator if the integrity verification fails for at least one segment.

43. The method of claim 41, further comprising:

selecting a ciphertext block of the secret random number from said string presented for decryption;

deciphering the selected ciphertext block to obtain the secret random number.

44. The method as defined in claim 43, performing said deciphering step with the inverse of a block cipher using a secret first key, said block cipher and said secret first key being the same as to those used at the message encryption method using the plurality of processors.

45. The method of claim 41, further comprising:

for the parallel encryption method generating said secret random number of ℓ bits in length by enciphering a counter initialized to a constant, said enciphering being done with said block cipher using said first key; and

incrementing after every different message encryption said counter by a number equal to a number of plaintext segments in the message; and

further comprising for decryption of the ciphertext segments of the partitioned ciphertext string the steps of:

selecting a counter block holding the count of the counter from said string presented for decryption;

enciphering the selected counter block to obtain said secret random number.

46. The method as defined in claim 45, wherein said enciphering the counter block step comprises enciphering with the block cipher using the same key as that used for encryption using a plurality of processors.

47. An encryption program product for providing both data confidentiality and integrity for a message, comprising:

first code for receiving an input plaintext string comprising a message and padding it as necessary such that its length is a multiple of ℓ bits;

second code for partitioning the padded input plaintext string into a plurality of equal-size blocks of ℓ bits in length;

third code for creating an MDC block of ℓ bits in length that includes the result of applying a non-cryptographic Manipulation Detection Code (MDC) function to the plurality of said equal-size blocks;

fourth code for making one and only one processing pass with a single cryptographic primitive over each of the said equal-size blocks and the MDC block to create a plurality of hidden ciphertext blocks each of ℓ bits in length; and

fifth code for performing a randomization function over said plurality of hidden ciphertext blocks to create a plurality of output ciphertext blocks each of ℓ bits in length.

48. The program product as defined in claim 47, comprising:

wherein said fourth code for making one and only one processing pass step comprises code for processing each of said equal-size blocks and the MDC block by an encryption scheme that is confidentiality-secure against chosen-plaintext attacks, wherein each of said equal-size blocks and the MDC block is processed by a block cipher using a first secret key (K) to obtain said plurality of hidden ciphertext blocks; and

wherein said fifth code for performing a randomization function comprises code for combining each of said hidden ciphertext blocks with a corresponding element of a sequence of unpredictable elements to create a set of output blocks of the ciphertext, wherein a hidden ciphertext block identified by an index i is combined with the element of the sequence identified by index i by an operation that has an inverse.

49. The program product as defined in claim 48, wherein said third code for creating an MDC block step comprises:

code for applying the non-cryptographic MDC function to the partitioned plaintext blocks; and

code for combining the result with a secret, ℓ-bit random vector generated on a per-message basis to obtain said MDC block.

50. A decryption program product that is the inverse of the encryption program product which provides both data confidentiality and integrity, comprising:

first code for presenting a string including ciphertext string for decryption;

second code for partitioning said ciphertext string into a plurality of ciphertext blocks comprising ℓ bits each;

third code for selecting n+1 ciphertext blocks from said plurality of ciphertext blocks representing n data blocks and one MDC block and performing a reverse randomization function on each of the selected n+1 ciphertext blocks to obtain a plurality of hidden ciphertext blocks each of ℓ bits in length;

fourth code for making one and only one processing pass with a single cryptographic primitive that is the inverse of an encryption single cryptographic primitive over the plurality of hidden ciphertext block to obtain a plurality of plaintext blocks comprising ℓ bits each;

fifth code for verifying integrity of the plaintext blocks using a non-cryptographic Manipulation Detection Code (MDC) function; and

sixth code for outputting the plurality of plaintext blocks as an accurate plaintext string if the integrity verification passes; and

seventh code for outputting a failure indicator if the integrity verification fails.

51. The program product as defined in claim 50, wherein said third code for performing said reverse randomization function comprises:

code for deriving a secret random number from said ciphertext string presented for decryption;

code for generating a sequence of unpredictable elements each of ℓ-bit length from said secret random number in the same manner as used at an encryption program product;

code for selecting n+1 ciphertext blocks from said plurality of ciphertext blocks representing n data blocks and one MDC block in the same order as that used at an encryption program product, and combining said selected ciphertext blocks with said sequence of unpredictable elements to obtain a plurality of hidden ciphertext blocks (z_i), such that each of the n+1 ciphertext blocks identified by index i is combined with the element of the sequence of unpredictable elements identified by index i, by the inverse of said operation used at the encryption program product;

wherein said fourth code for making one and only one processing pass comprises code for decrypting the plurality of hidden ciphertext blocks with the inverse of the block cipher used at an encryption program product with a first secret key (K), the result of the decryption being a plurality of n decrypted plaintext data blocks and one decrypted MDC block each of ℓ-bit length; and

wherein said fifth code for verifying integrity step comprises code for creating an MDC decryption block by applying the non-cryptographic Manipulation Detection Code function to the n decrypted plaintext data blocks and combining the result with a secret, ℓ-bit random vector , said combining operation being the same as the combining operation at the encryption program product, and said secret random vector being derived from said secret random number in the same manner as at the encryption program product; and comparing said created MDC decryption block with the decrypted MDC block.

52. An encryption system for providing both data confidentiality and integrity for a message, comprising:

a first component for receiving an input plaintext string comprising a message and padding it as necessary such that its length is a multiple of ℓ bits;

a second component for partitioning the padded input plaintext string into a plurality of equal-size blocks of ℓ bits in length;

a third component for creating an MDC block of ℓ bits in length that includes the result of applying a non-cryptographic Manipulation Detection Code (MDC) function to the plurality of said equal-size blocks;

a fourth component for making one and only one processing pass with a single cryptographic primitive over each of the said equal-size blocks and the MDC block to create a plurality of hidden ciphertext blocks each of ℓ bits in length; and

a fifth component for performing a randomization function over said plurality of hidden ciphertext blocks to create a plurality of output ciphertext blocks each of ℓ bits in length.

53. The system as defined in claim 52, wherein said fourth component for making one and only one processing pass step comprises a component for processing each of said equal-size blocks and the MDC block by an encryption scheme that is confidentiality-secure against chosen-plaintext attacks, wherein each of said equal-size blocks and the MDC block is processed by a block cipher using a first secret key to obtain said plurality of hidden ciphertext blocks; and

wherein said fifth component for performing a randomization function comprises a component for combining each of said hidden ciphertext blocks with a corresponding element of a sequence of unpredictable elements to create a set of output blocks of the ciphertext, wherein a hidden ciphertext block identified by an index i is combined with the element of the sequence identified by index i by an operation that has an inverse.

54. The system as defined in claim 53, wherein said third component for creating an MDC block step comprises:

a component for applying the non-cryptographic MDC function to the partitioned plaintext blocks; and

a component for combining the result with a secret, ℓ-bit random vector generated on a per-message basis to obtain said MDC block.

55. A decryption system that is the inverse of the encryption system which provides both data confidentiality and integrity, comprising:

a first component for presenting a string including ciphertext string for decryption;

a second component for partitioning said ciphertext string into a plurality of ciphertext blocks comprising ℓ bits each;

a third component for selecting n+1 ciphertext blocks from said plurality of ciphertext blocks representing n data blocks and one MDC block and performing a reverse randomization function on each of the selected n+1 ciphertext blocks to obtain a plurality of hidden ciphertext blocks each of ℓ bits in length;

a fourth component for making one and only one processing pass with a single cryptographic primitive that is the inverse of an encryption single cryptographic primitive over the plurality of hidden ciphertext block to obtain a plurality of plaintext blocks comprising ℓ bits each;

a fifth component for verifying integrity of the plaintext blocks using a non-cryptographic Manipulation Detection Code (MDC) function; and

a sixth component for outputting the plurality of plaintext blocks as an accurate plaintext string if the integrity verification passes; and

a seventh component for outputting a failure indicator if the integrity verification fails.

56. The system as defined in claim 55, wherein said third component for performing said reverse randomization function comprises:

a component for deriving a secret random number from said ciphertext string presented for decryption;

a component for generating a sequence of unpredictable elements each of ℓ-bit length from said secret random number in the same manner as used at an encryption system;

a component for selecting n+1 ciphertext blocks from said plurality of ciphertext blocks representing n data blocks and one MDC block in the same order as that used at an encryption system, and combining said selected ciphertext blocks with said sequence of unpredictable elements to obtain a plurality of hidden ciphertext blocks, such that each of the n+1 ciphertext blocks identified by index i is combined with the element of the sequence of unpredictable elements identified by index i, by the inverse of said operation used at the encryption system;

wherein said fourth component for making one and only one processing pass comprises a component for decrypting the plurality of hidden ciphertext blocks with the inverse of the block cipher used at an encryption system with a first secret key (K), the result of the decryption being a plurality of n decrypted plaintext data blocks and one decrypted MDC block each of ℓ-bit length; and

wherein said fifth component for verifying integrity step comprises a component for creating an MDC decryption block by applying the non-cryptographic Manipulation Detection Code function to the n decrypted plaintext data blocks and combining the result with a secret, ℓ-bit random vector, said combining operation being the same as the combining operation at the encryption system, and said secret random vector being derived from said secret random number in the same manner as at the encryption system; and comparing said created MDC decryption block with the decrypted MDC block.

57. A program product for parallel encryption processing of a message comprising:

first code for partitioning said input plaintext string into a plurality of input plaintext segments;

second code for concurrently presenting each different one of said plurality of input plaintext segments to a different one of a plurality of encryption processors, each of said different processors using a different ℓ-bit secret random number per segment to obtain a ciphertext segment using an encryption code providing both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, and using a non-cryptographic Manipulation Detection Code function, wherein said single cryptographic primitive is a ℓ-bit block cipher using a secret first key;

third code for assembling the plurality of ciphertext segments into a ciphertext string; and

fourth code for outputting the ciphertext string.

58. The program product as defined in claim 57, wherein said third code for assembling comprises code for including in the ciphertext string the number of ciphertext segments, a ciphertext segment index, a length of each ciphertext segment and a sequence of ciphertext segments.

59. A program product for parallel decryption processing of a message comprising:

first code for presenting a string including the ciphertext string of a message for decryption;

second code for partitioning said ciphertext string into a plurality of ciphertext segments;

third code for concurrently presenting said plurality of ciphertext segments to a plurality of processors;

fourth code for obtaining a different secret random number per ciphertext segment from a secret random number in the same manner as at the parallel encryption program product;

fifth code for decrypting each ciphertext segment using said different secret random number per ciphertext segment to obtain a plaintext segment, using a decryption method that is the inverse of an encryption method used in the parallel encryption method that provides both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, wherein said single cryptographic primitive is a ℓ-bit block cipher using a secret first key, and using a non-cryptographic

Manipulation Detection Code function for verifying integrity of the plaintext blocks of each plaintext segment;

sixth code for assembling the plurality of plaintext segments into a plaintext string; and

seventh code for verifying the integrity of the plaintext segments and their sequence and outputting the plaintext string if the integrity verification passes.

60. The program product as defined in claim 59, further comprising code for outputting a failure indicator if the integrity verification fails for at least one segment.

61. A system for parallel encryption processing of a message comprising:

a first component for partitioning said input plaintext string into a plurality of input plaintext segments;

a second component for concurrently presenting each different one of said plurality of input plaintext segments to a different one of a plurality of encryption processors, each of said different processors using a different ℓ-bit secret random number per segment to obtain a ciphertext segment using an encryption component providing both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, and using a non-cryptographic Manipulation Detection Code function, wherein said single cryptographic primitive is a ℓ-bit block cipher using a secret first key;

a third component for assembling the plurality of ciphertext segments into a ciphertext string; and

a fourth component for outputting the ciphertext string.

62. The system as defined in claim 61, wherein said third component for assembling comprises a component for including in the ciphertext string the number of ciphertext segments, a ciphertext segment index, a length of each ciphertext segment and a sequence of ciphertext segments.

63. A system for parallel decryption processing of a message comprising:

a first component for presenting a string including the ciphertext string of a message for decryption;

a second component for partitioning said ciphertext string into a plurality of ciphertext segments;

a third component for concurrently presenting said plurality of ciphertext segments to a plurality of processors;

a fourth component for obtaining a different secret random number per ciphertext segment from a secret random number in the same manner as at the parallel encryption system;

a fifth component for decrypting each ciphertext segment using said different secret random number per ciphertext segment to obtain a plaintext segment, using a decryption method that performs the inverse operation of an encryption method used in the parallel encryption method that provides both data confidentiality and integrity with a single processing pass over the input plaintext segment and a single cryptographic primitive, wherein said single cryptographic primitive is a ℓ-bit block cipher using a secret first key, and using a non-cryptographic Manipulation Detection Code function for verifying integrity of the plaintext blocks of each plaintext segment;

a sixth component for assembling the plurality of plaintext segments into a plaintext string; and

a seventh component for verifying the integrity of the plaintext segments and their sequence and outputting the plaintext string if the integrity verification passes.

64. The system as defined in claim 63, further comprising a component for outputting a failure indicator if the integrity verification fails for at least one segment.